Université d'Ottawa
Faculté de génie

École de science
d'informatique
et de génie électrique

uOttawa
L'Université canadienne
Canada's university

University of Ottawa
Faculty of Engineering

School of Electrical
Engineering
and Computer Science

Assignment 3
CSI2120 Programming Paradigms

Winter 2017

Due on March 13th, 2017 before 11:00 pm in Virtual Campus

[5 marks in total]

All code must be handed-in as source files!

PDFs, screenshots, word documents etc. will receive an automatic 0.

Question 1. **[3.5 marks]**

A database for student and course information system needs to be constructed. The following facts are given:

```
% name, studentId, course list
student(name(blake, [ann]), 33333, ['CSI2120'] ).

% course, type, name as list of text, maximum marks
evaluation('CSI2120', assignment(1),  ['Prolog', database ],  5).

% course, studentId, evaluation, mark
mark('CSI2120', 33333, assignment(1), 3.5 ).
```

a) Design a predicate `addStudent/0` which interactively accepts input to create a new student/3 record with an empty class list. Your predicate must check that the student is not yet in the system and the student ID is not yet used. Consider the following example:

```
?- addStudent.
Student last name: doe.
Student first name: [jim,k].
Student Id: 23123.
true.
```

b) Design a predicate `add/2` which given a course and student ID, adds a course to this student's course list. Your predicate must ensure that the same course can only be added once. Consider the following example:

```
?- add('CSI2120',23123).
true.
```

c) Design a predicate `addAllMarks/2` which given a course and evaluation, loops over all students in the course and let a user enter marks. Your predicate should loop over students who have the course in their course list and which have not yet received a mark for the evaluation. Consider the following example (user input in italic):

```
?- addAllMarks('CSI2120',midterm(1)).
name(blake,[ann]) Mark (out of 26): 25.
name(doe,[jane,j]) Mark (out of 26): 22.
name(doe,[jim,k]) Mark (out of 26): 14.
false.
```

Note, the information provided to the user in input. Also, note that your program must check if an entered mark is valid. For example:

```
?- addAllMarks('CSI2120',assignment(1)).
name(doe,[jane,j]) Mark (out of 5): 7.
name(doe,[jane,j]) Mark (out of 5): -2.
name(doe,[jane,j]) Mark (out of 5): 3.
name(doe,[jim,k]) Mark (out of 5): 4.
false.
```

d) Design a predicate `listAllMarks/3` which given a course and evaluation, shows the student number and mark of all students which completed the evaluation. Use `setof` to achieve the result. For example:

```
?- listAllMarks('CSI2120',midterm(1),L).
L = [ (23123, 14), (33333, 25), (88345, 22)].
```

e) Create a predicate `averageMark/3` that computes the average mark of a given evaluation. Hint: Create a list of all marks and create a helper predicate to sum the marks. For example:

```
?- averageMark('CSI2120',midterm(1),A).
% A = 20.333333333333332.
```

Question 2. **[1.5 marks]**

Write a predicate `cleanListDCG/3` which parses a list and returns a new list with the same numbers but all elements that are not a number removed. You must use only DCG notation. Your implementation of `cleanListDCG/3` must work correctly when called from `cleanList/2`.

```
cleanList(L,LL) :- cleanListDCG(LL,L,[]),!.
```

Example:

```
?- cleanList([1,2,d,67,3.2,'CSI2120',foo,5],LL).
LL = [1, 2, 67, 3.2, 5].
```